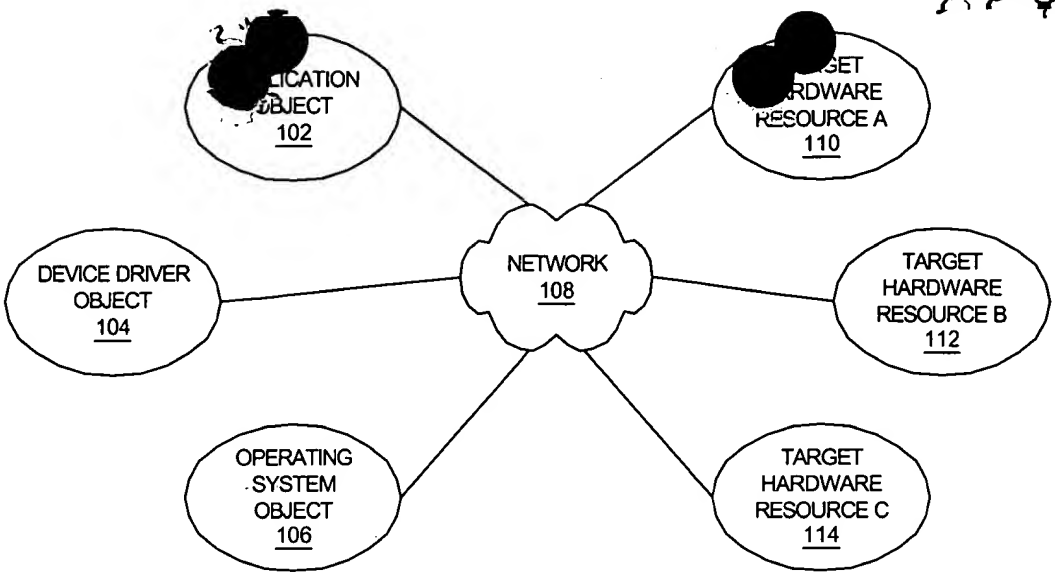

Prior Art

Figure 1

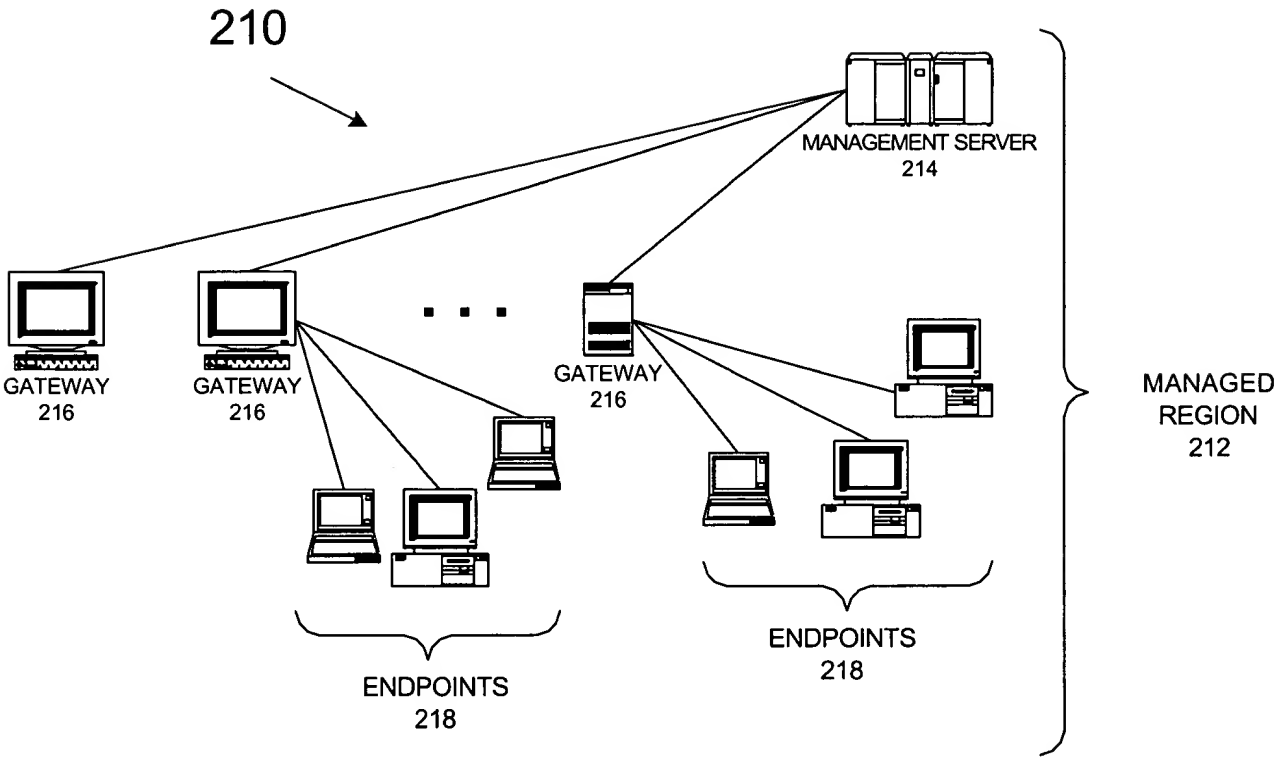
210
MANAGEMENT SERVER
214
GATEWAY
216
GATEWAY
216
GATEWAY
216
ENDPOINTS
218
ENDPOINTS
218
MANAGED
REGION
212

Figure 2A

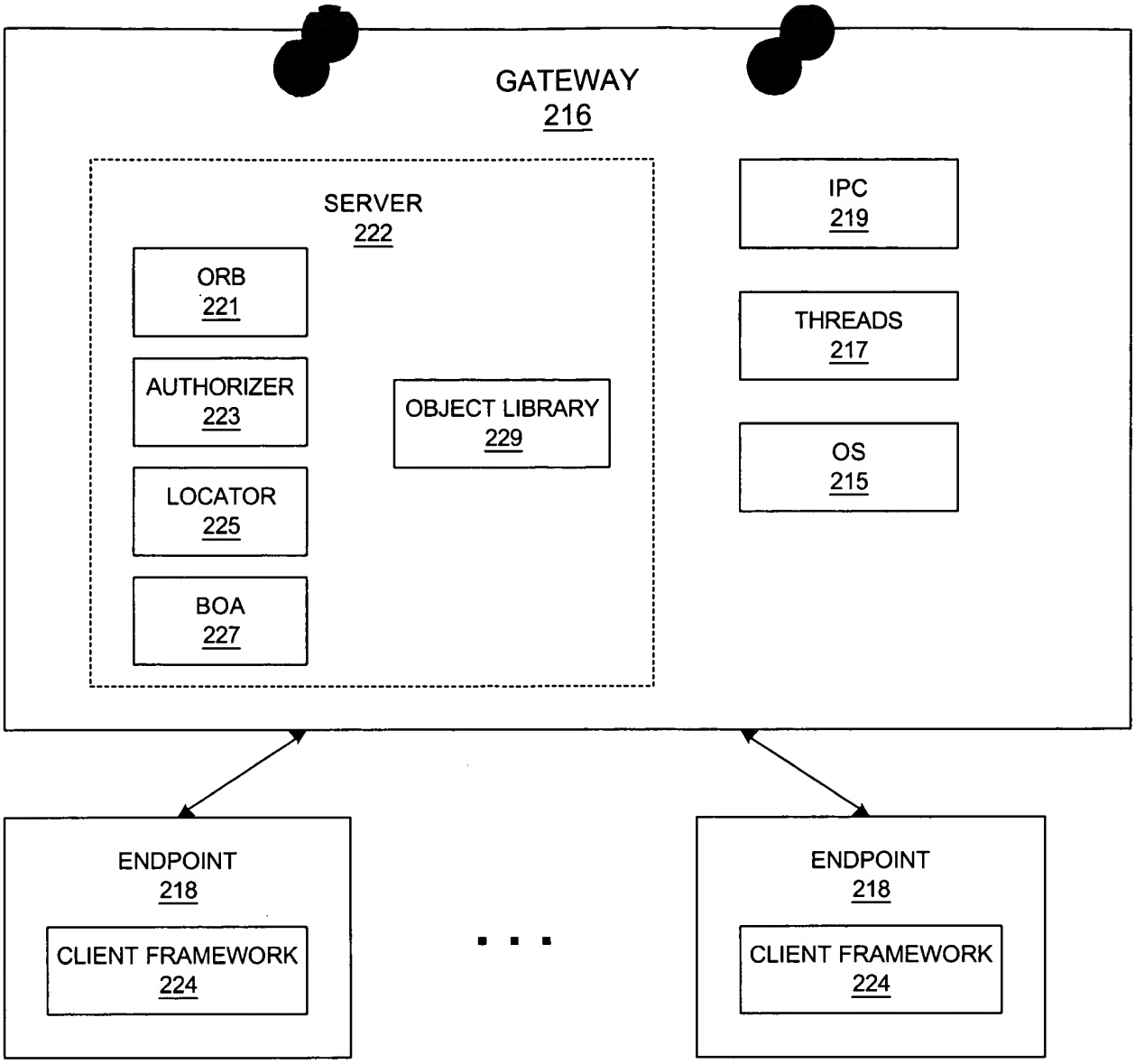

Figure 2B

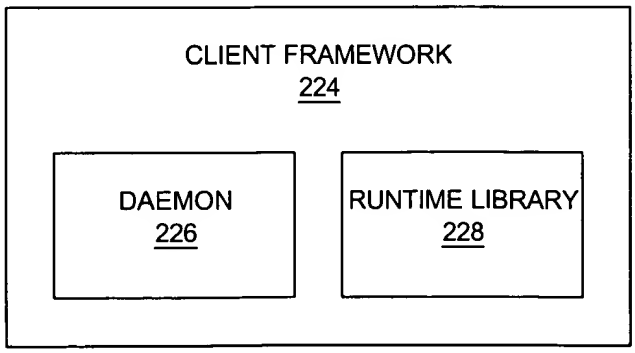

Figure 2C

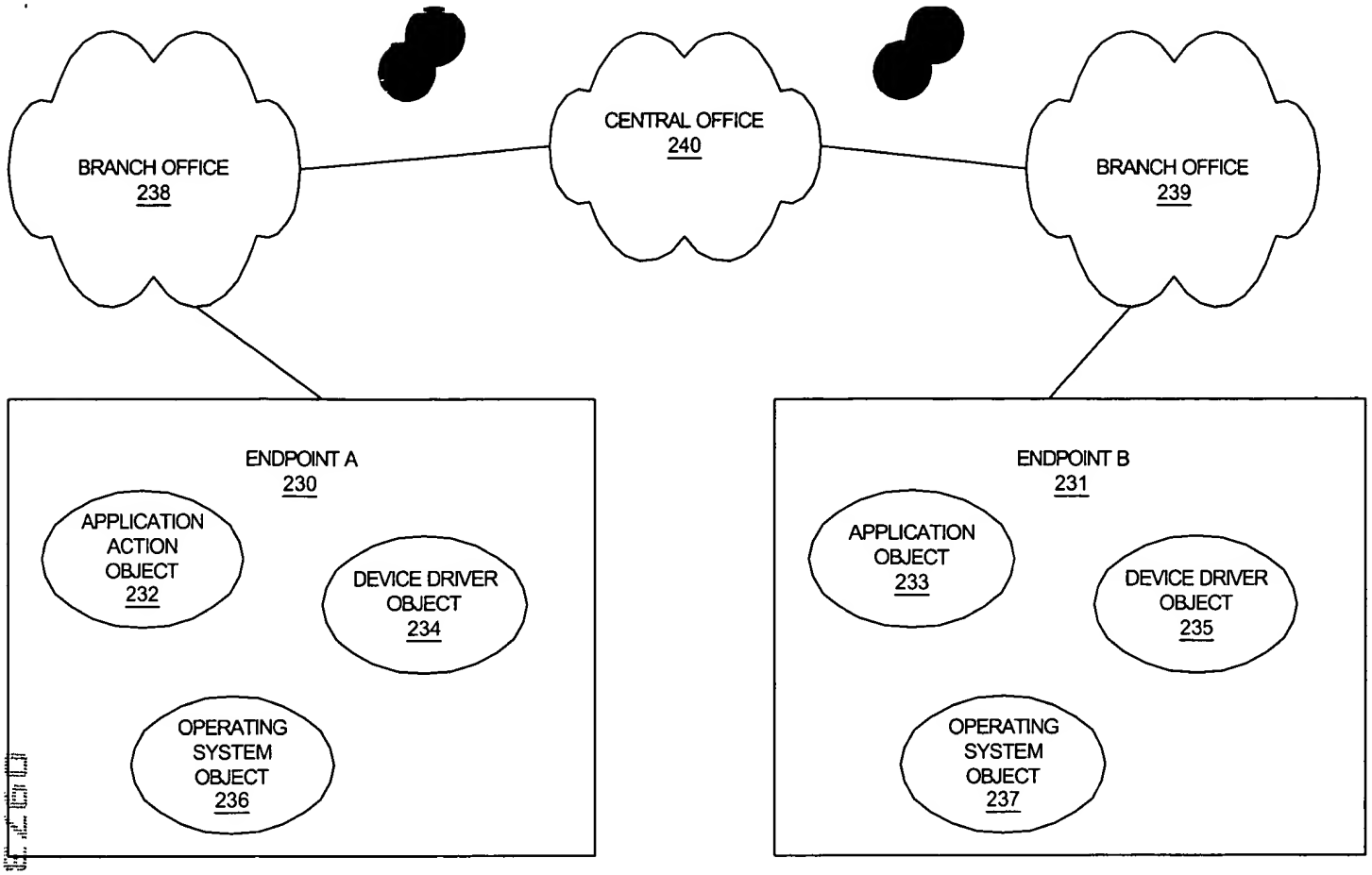

Figure 2D

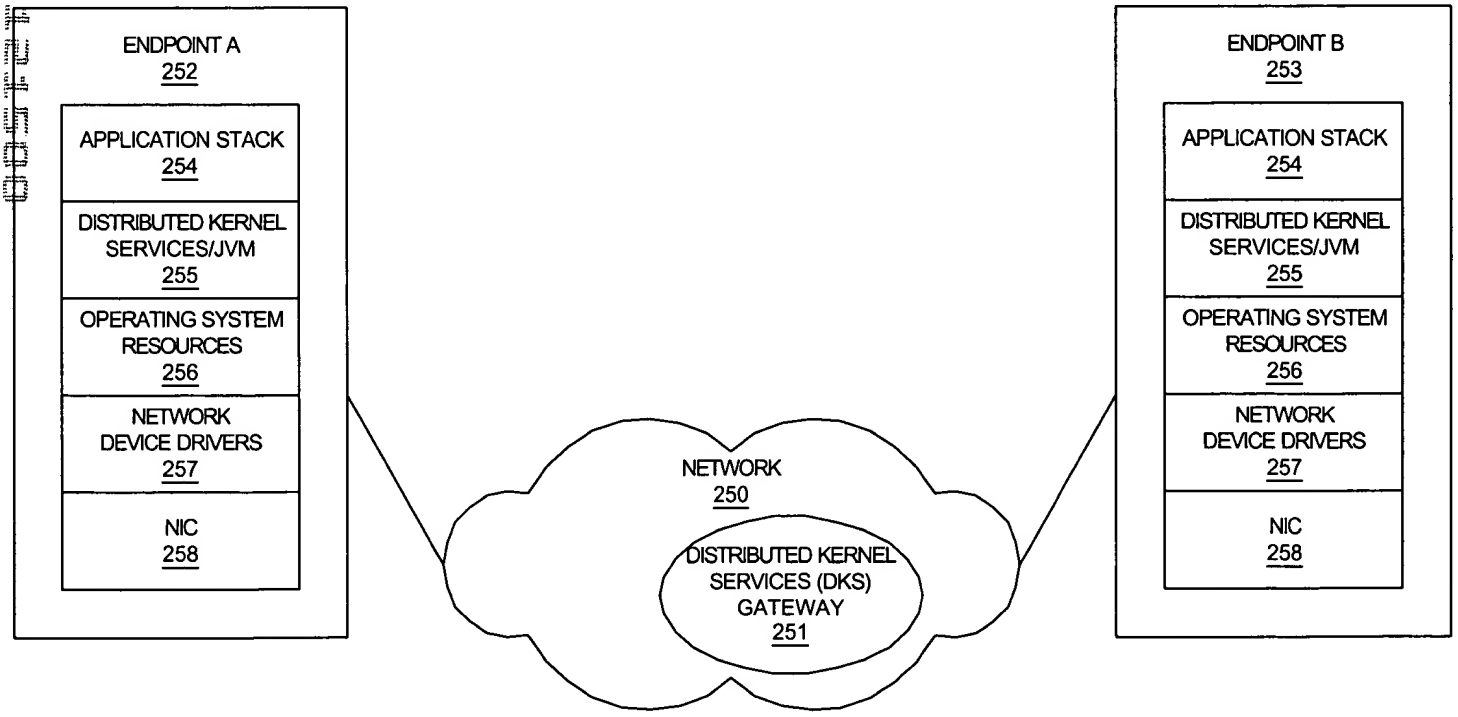

Figure 2E

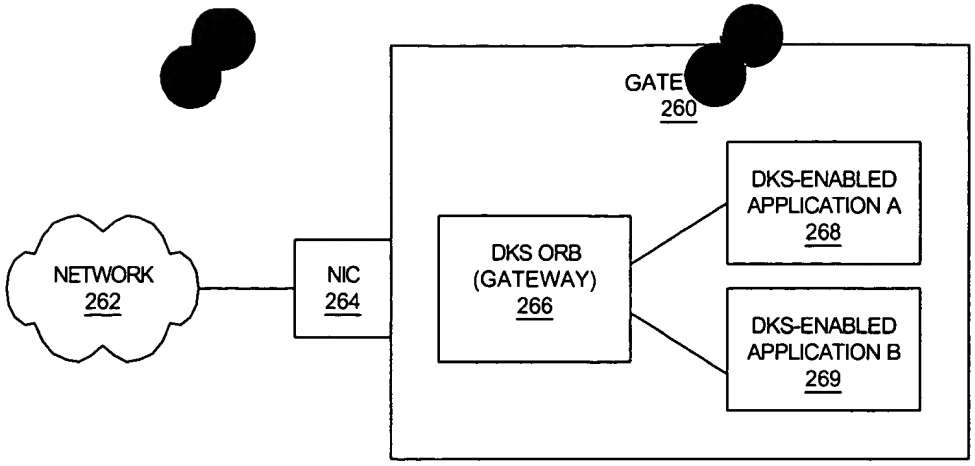

Figure 2F

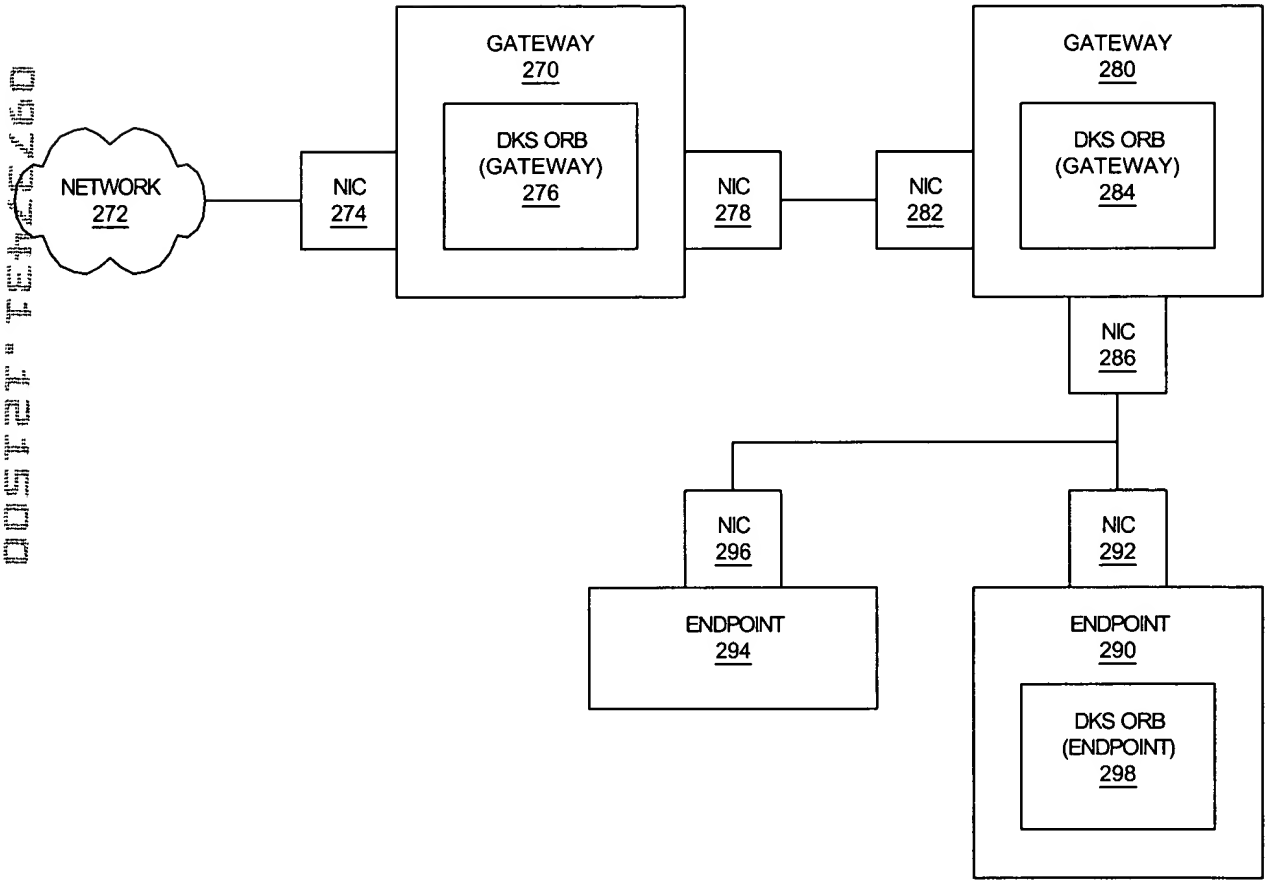

Figure 2G

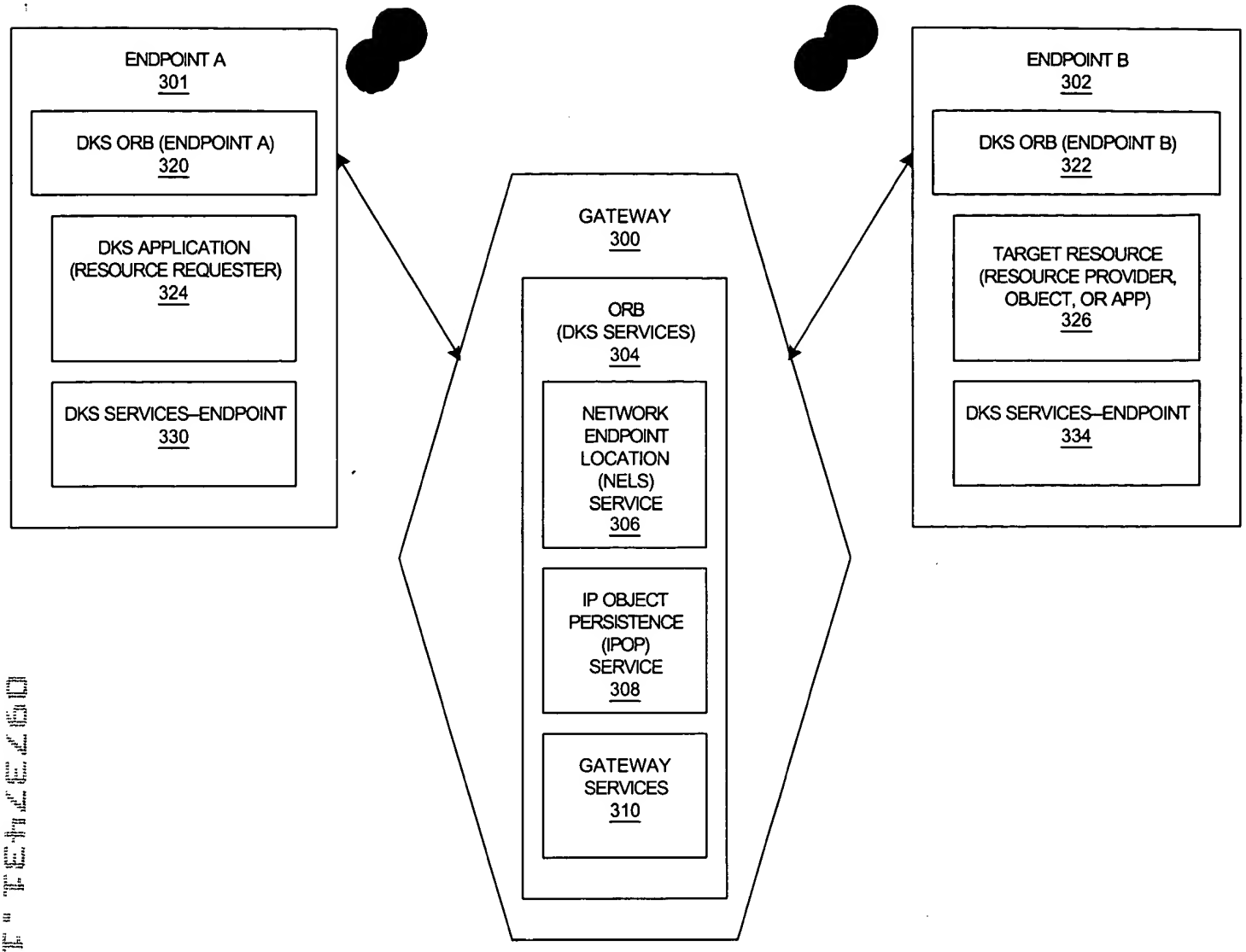

Figure 3

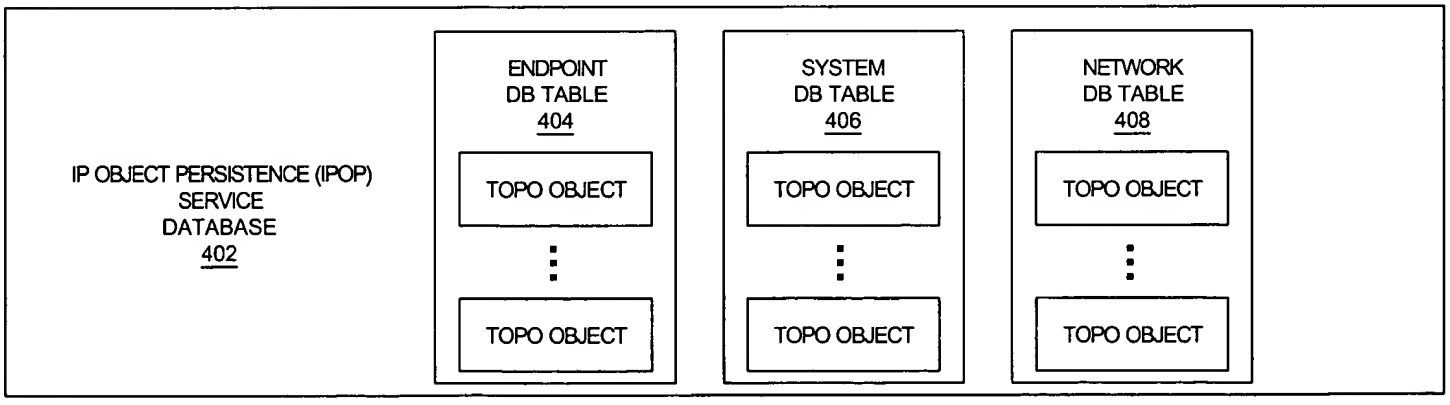

Figure 4

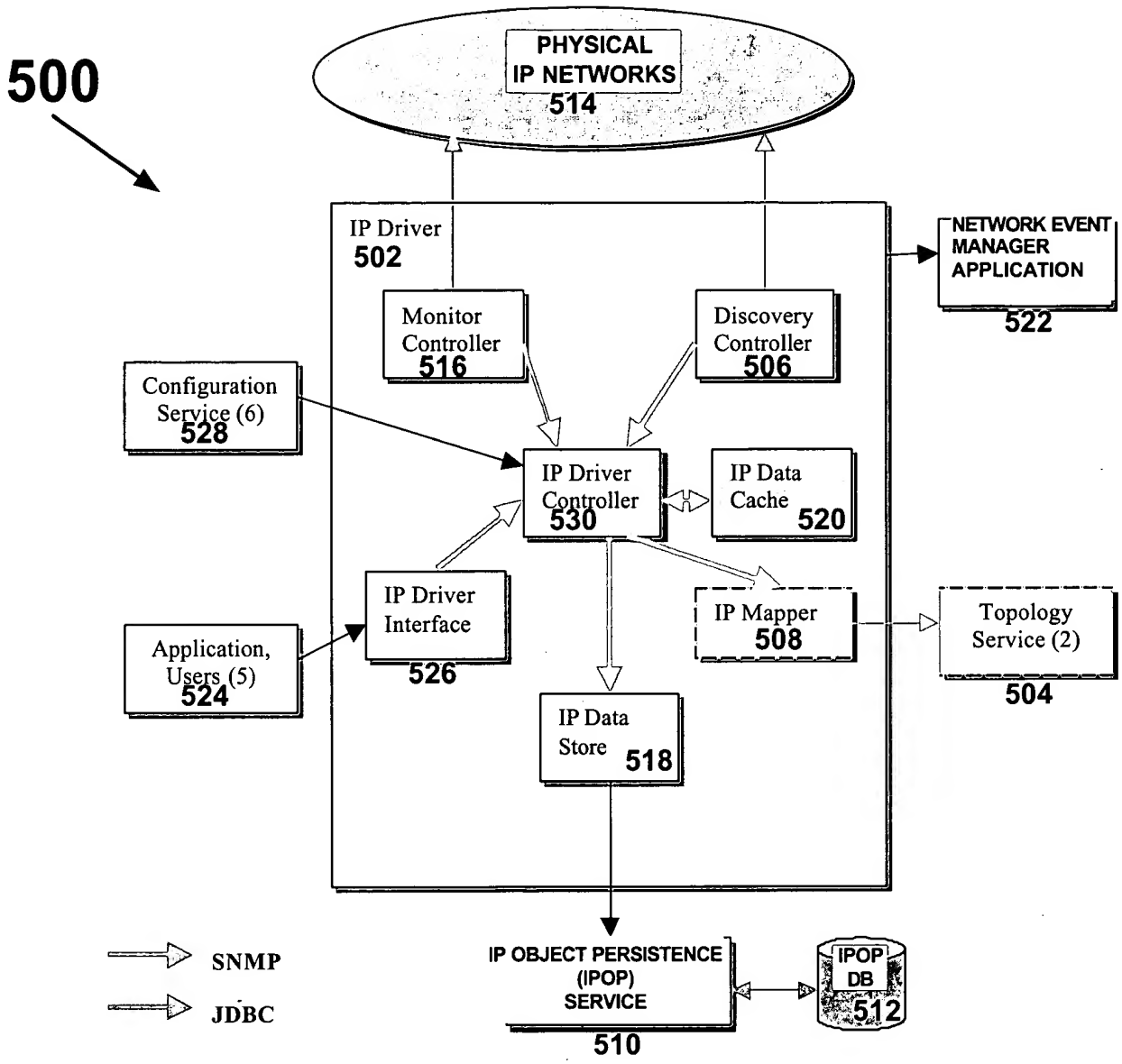
500
PHYSICAL IP NETWORKS 514
IP Driver 502
NETWORK EVENT MANAGER APPLICATION 522
Monitor Controller 516
Discovery Controller 506
Configuration Service (6) 528
IP Driver Controller 530
IP Data Cache 520
IP Driver Interface 526
IP Mapper 508
Topology Service (2) 504
Application, Users (5) 524
IP Data Store 518
SNMP
JDBC
IP OBJECT PERSISTENCE (IPOP) SERVICE 510
IPOP DB 512

Figure 5A

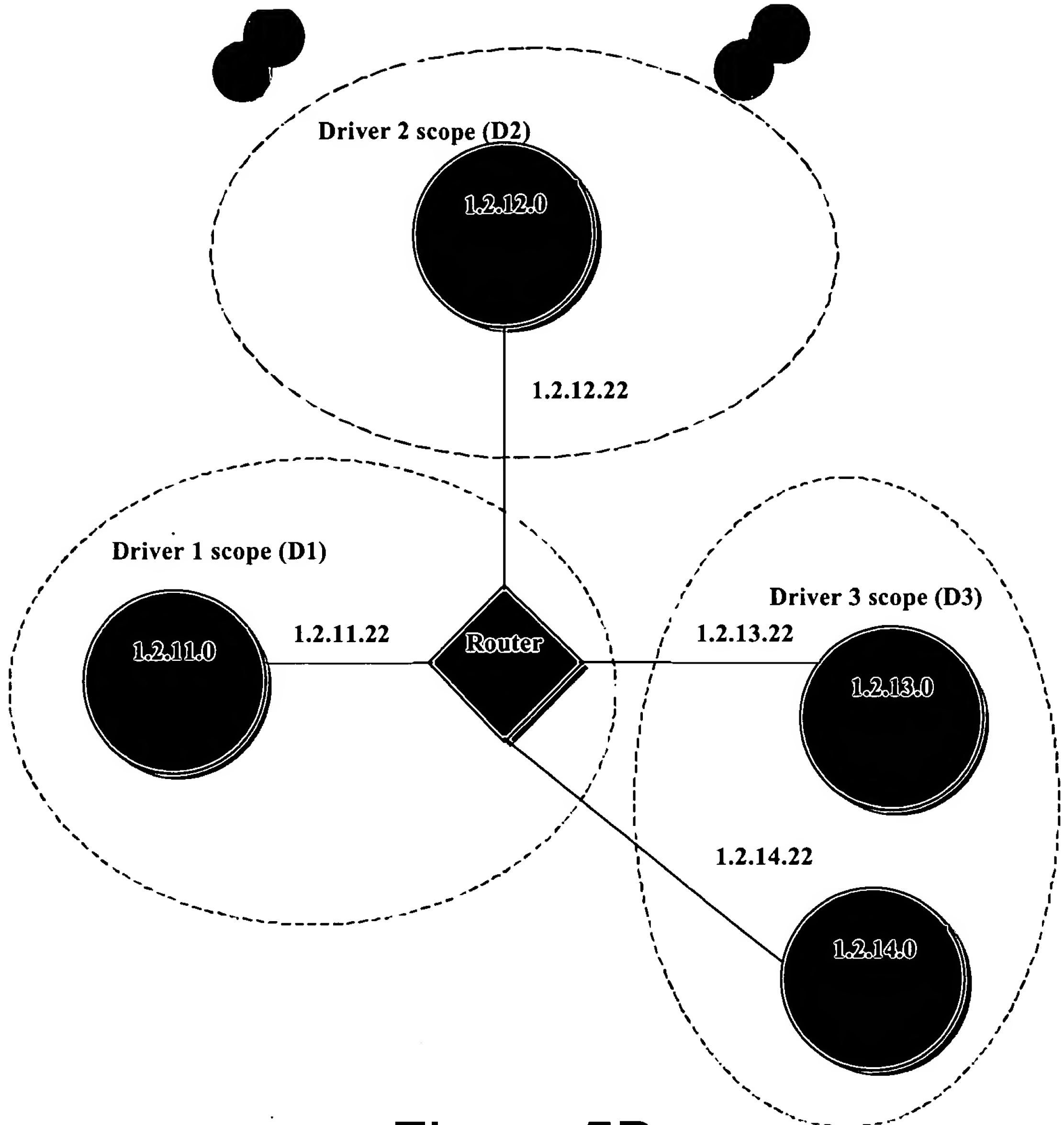

Figure 5B

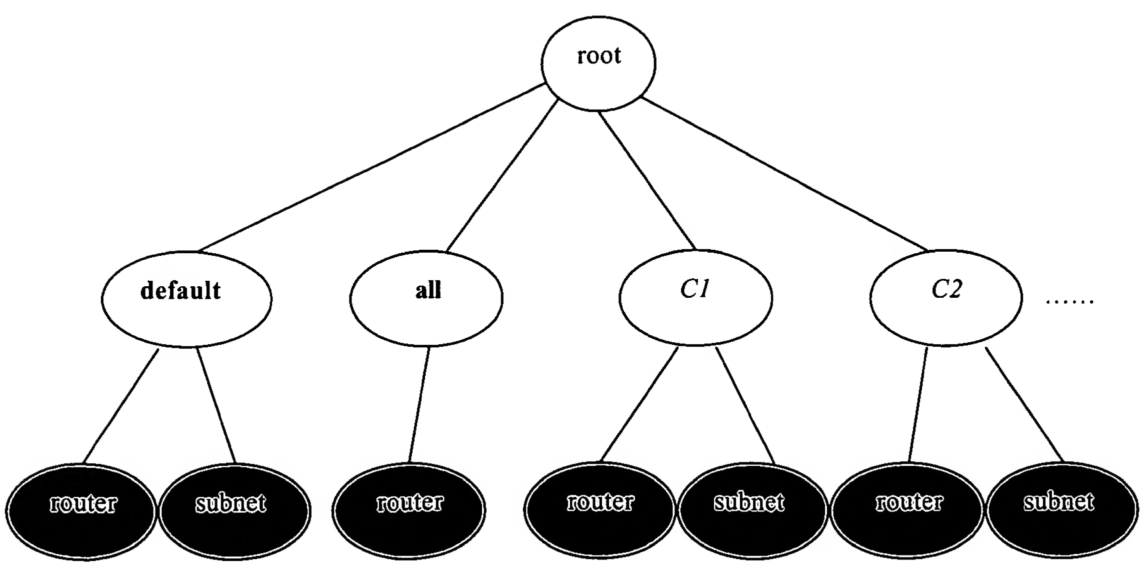

Figure 5C

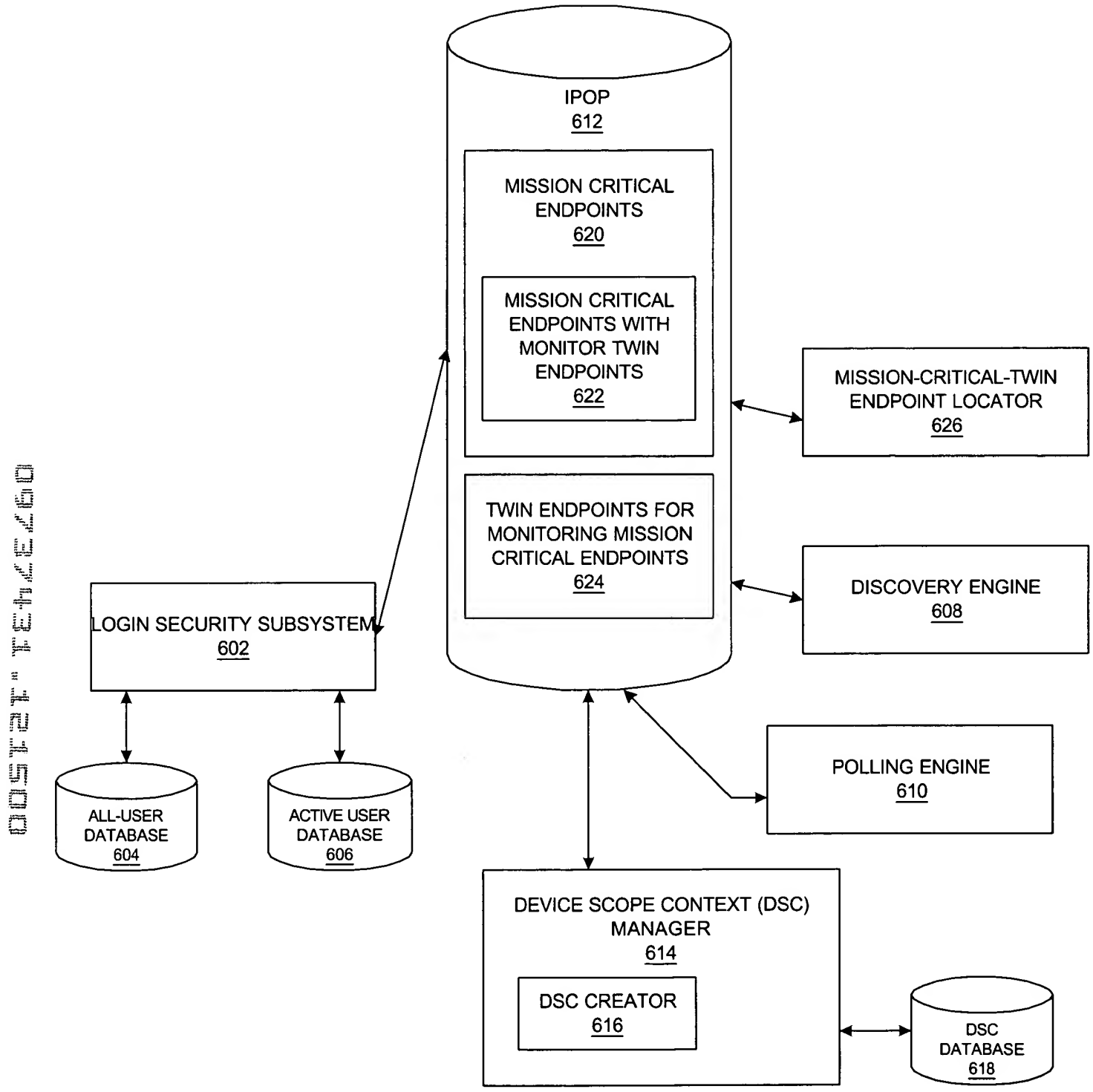
IPOP
612
MISSION CRITICAL ENDPOINTS
620
MISSION CRITICAL ENDPOINTS WITH MONITOR TWIN ENDPOINTS
622
TWIN ENDPOINTS FOR MONITORING MISSION CRITICAL ENDPOINTS
624
MISSION-CRITICAL-TWIN ENDPOINT LOCATOR
626
DISCOVERY ENGINE
608
POLLING ENGINE
610
LOGIN SECURITY SUBSYSTEM
602
ALL-USER DATABASE
604
ACTIVE USER DATABASE
606
DEVICE SCOPE CONTEXT (DSC) MANAGER
614
DSC CREATOR
616
DSC DATABASE
618

Figure 6

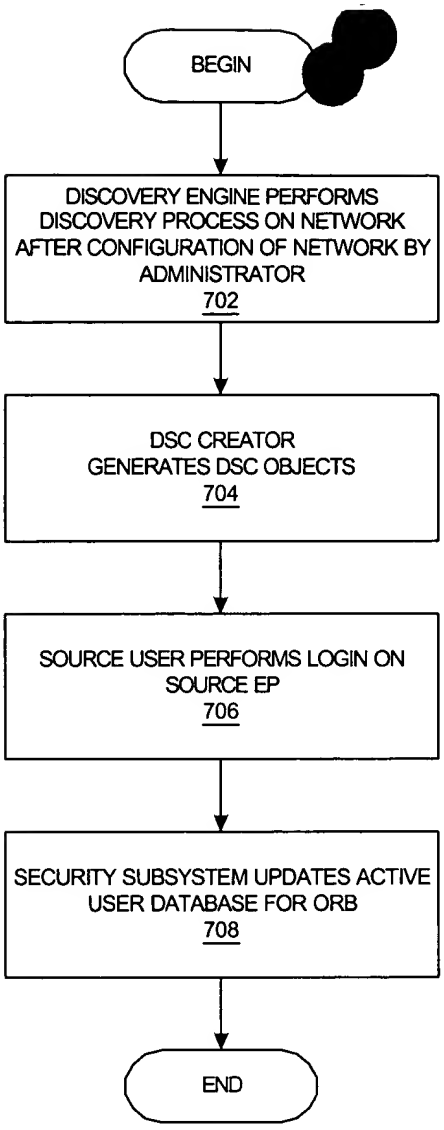

Figure 7A

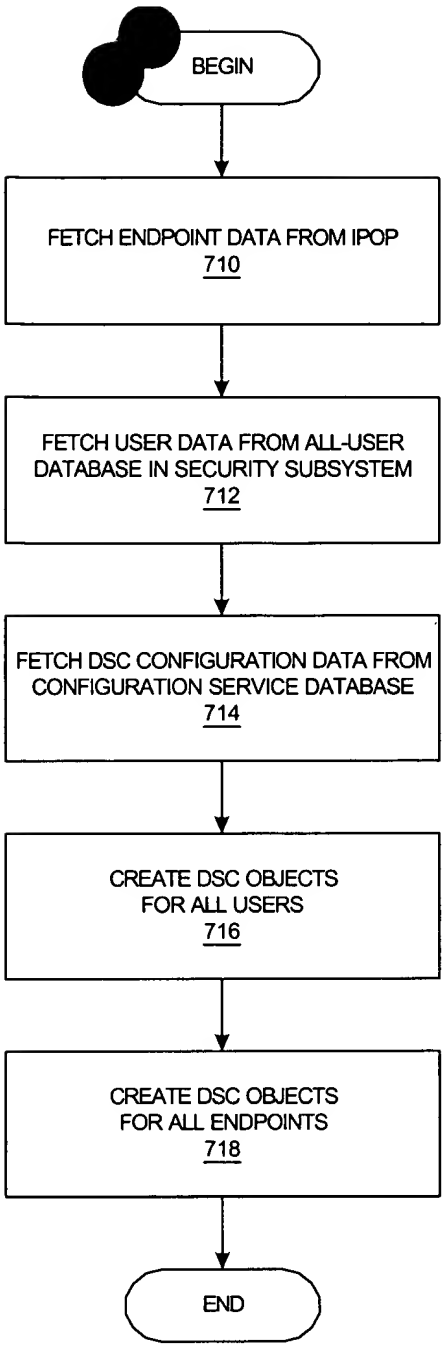

Figure 7B

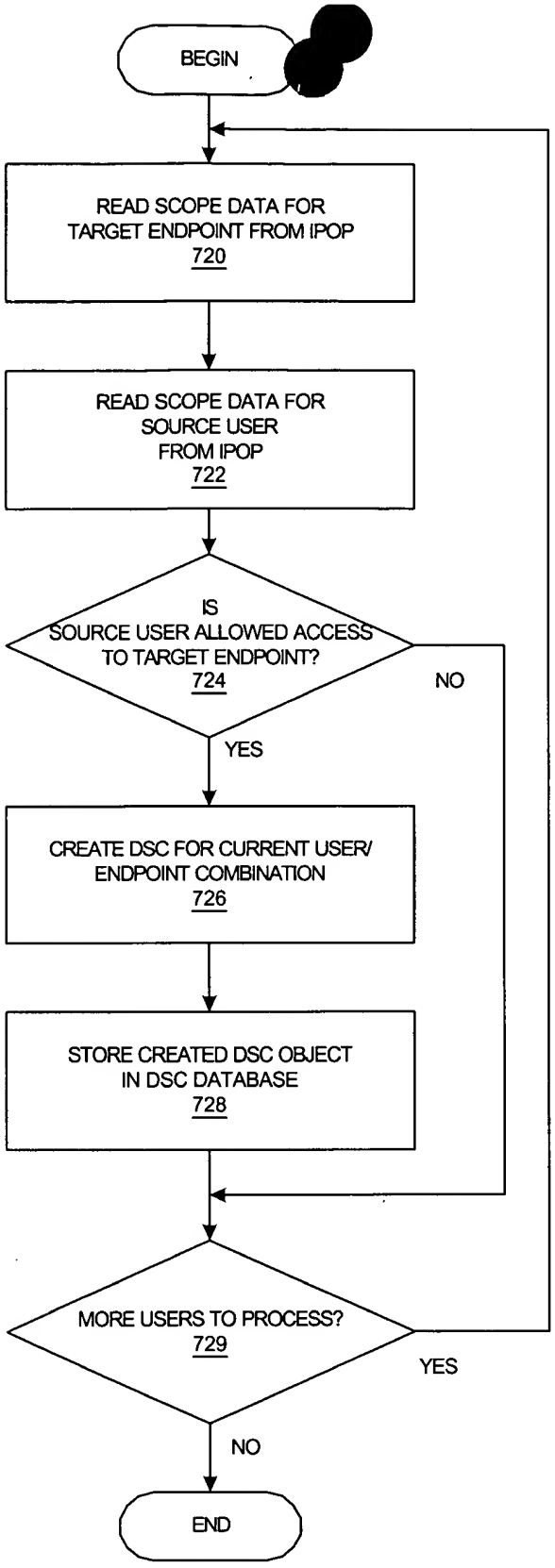

Figure 7C

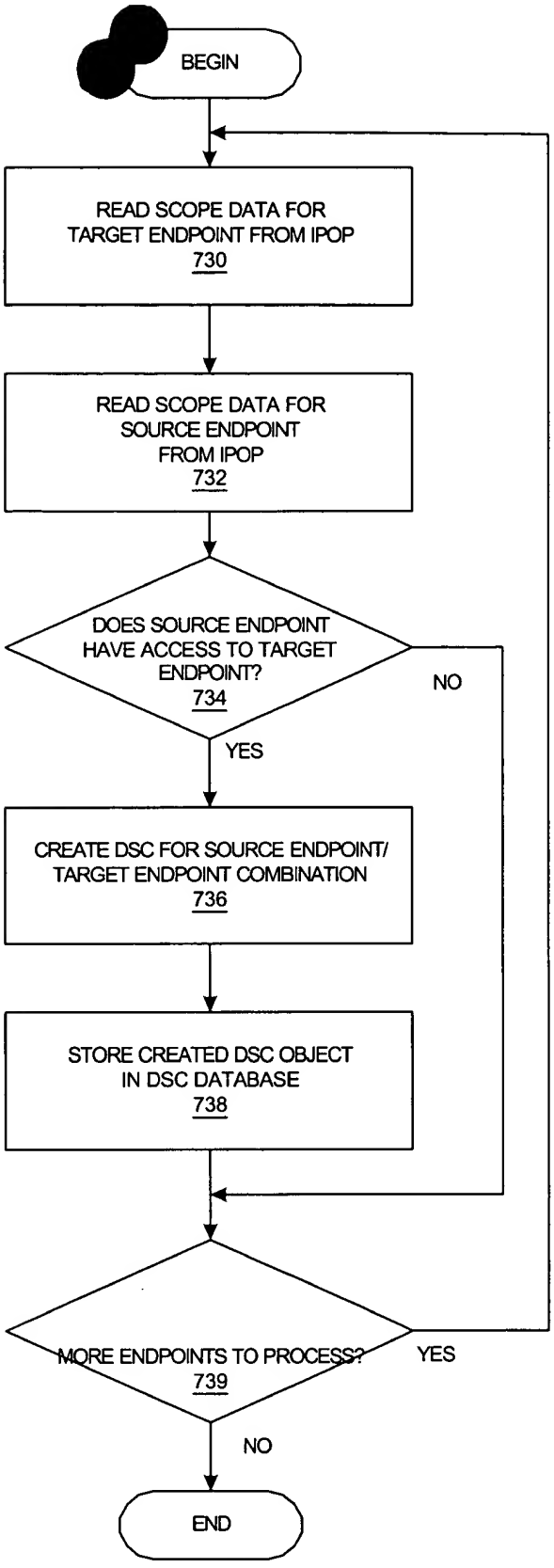

Figure 7D

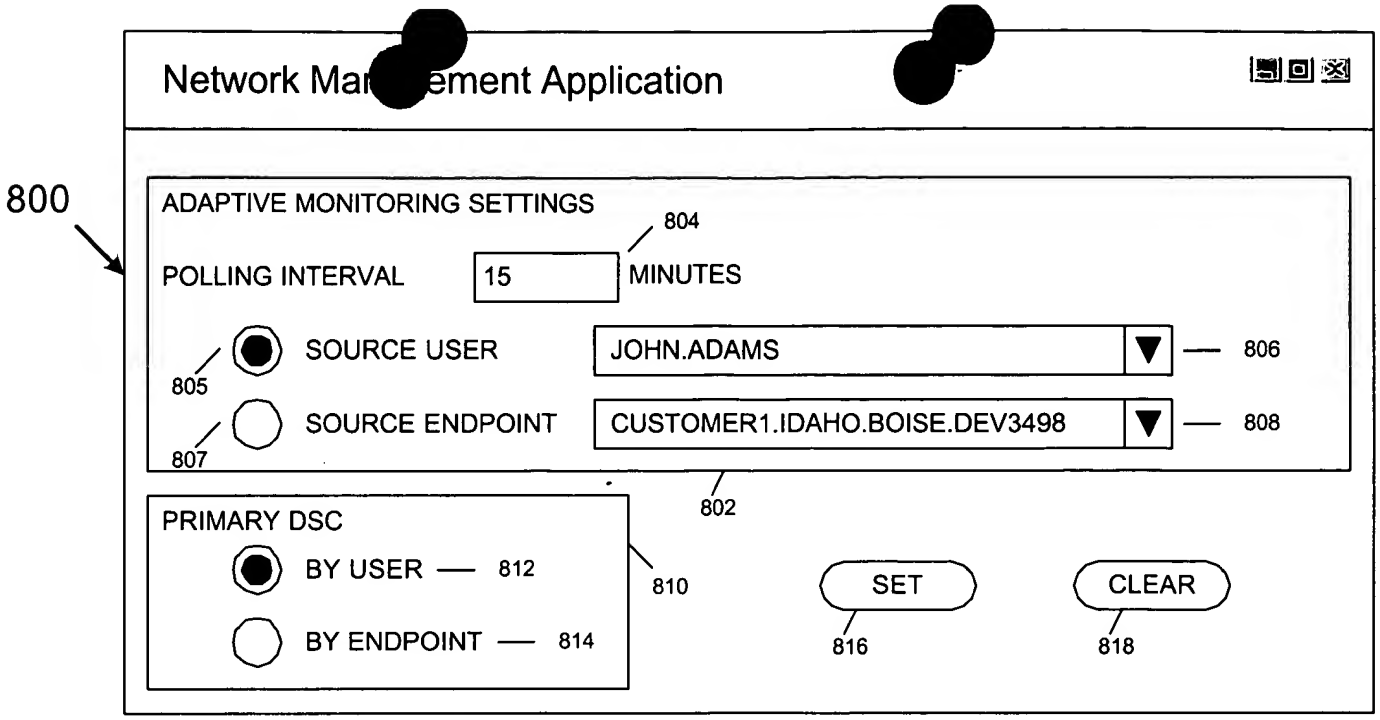

Figure 8A

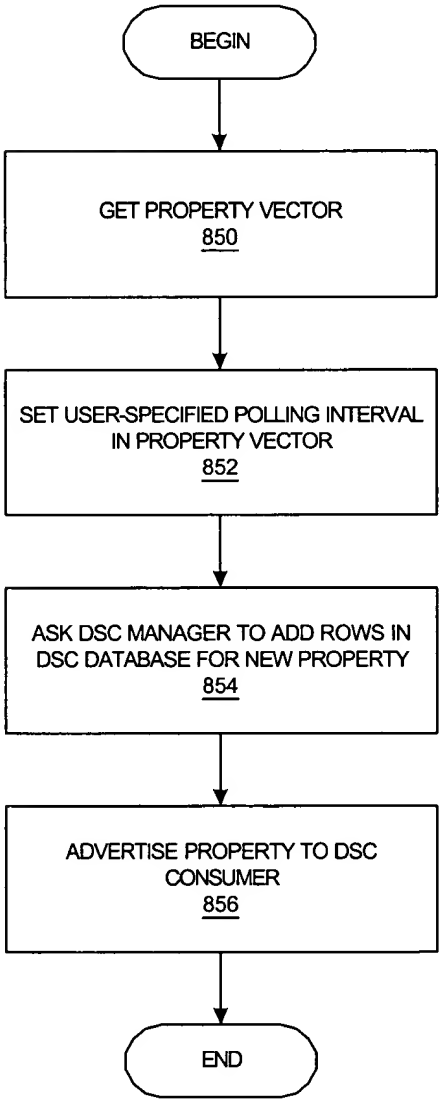

Figure 8C

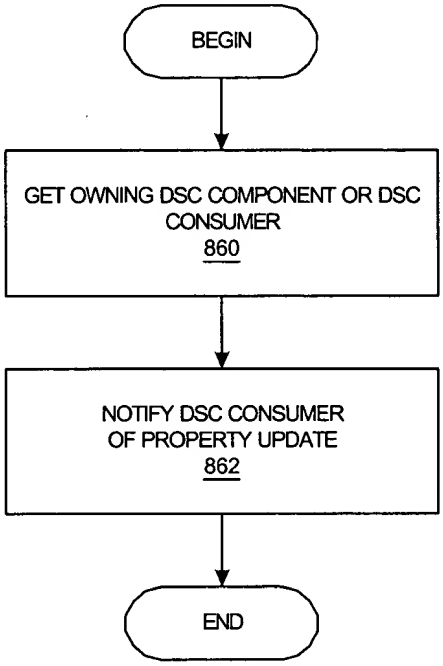

Figure 8D

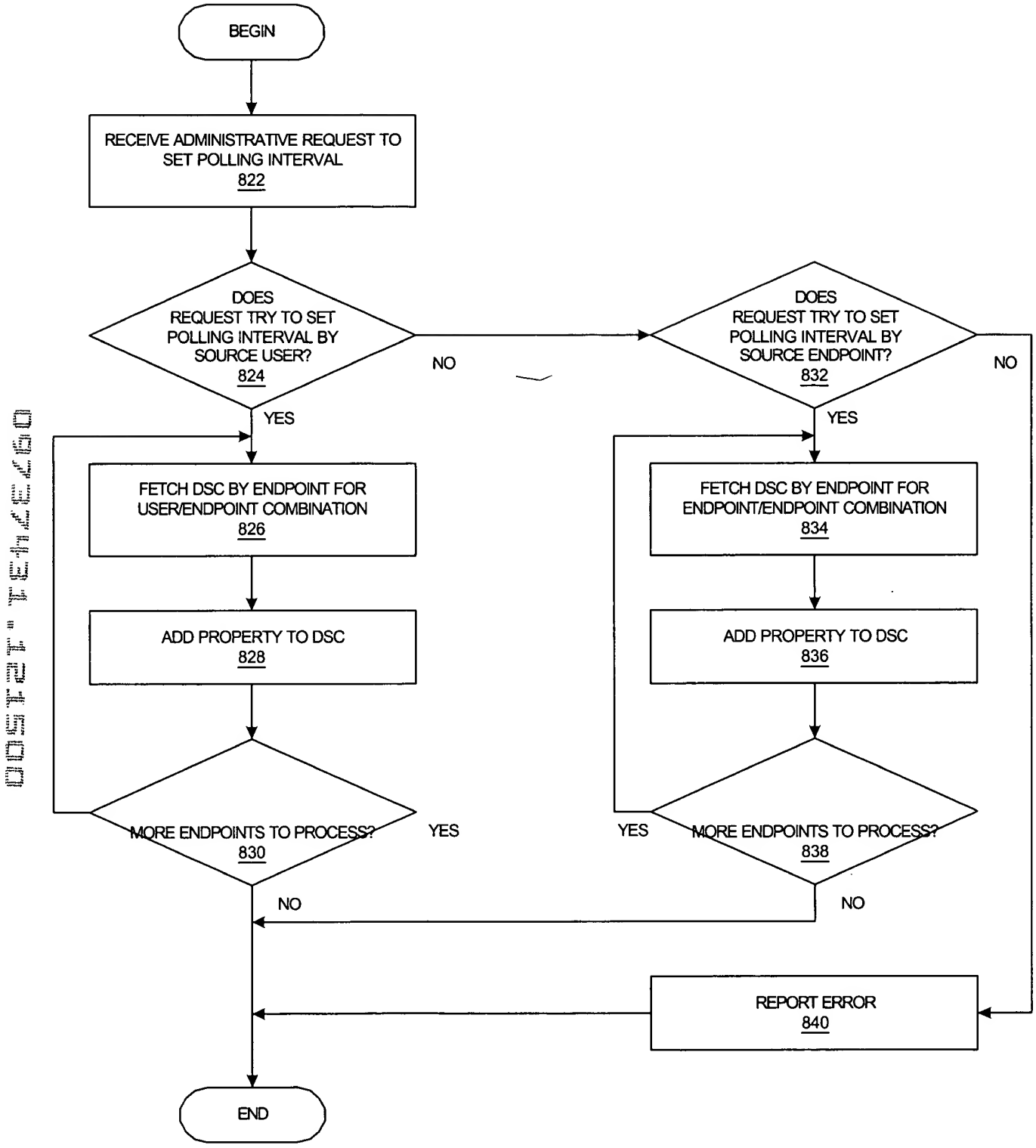
BEGIN
RECEIVE ADMINISTRATIVE REQUEST TO SET POLLING INTERVAL
822
DOES REQUEST TRY TO SET POLLING INTERVAL BY SOURCE USER?
824
NO
YES
DOES REQUEST TRY TO SET POLLING INTERVAL BY SOURCE ENDPOINT?
832
NO
YES
FETCH DSC BY ENDPOINT FOR USER/ENDPOINT COMBINATION
826
ADD PROPERTY TO DSC
828
MORE ENDPOINTS TO PROCESS?
830
YES
NO
FETCH DSC BY ENDPOINT FOR ENDPOINT/ENDPOINT COMBINATION
834
ADD PROPERTY TO DSC
836
MORE ENDPOINTS TO PROCESS?
838
YES
NO
REPORT ERROR
840
END

Figure 8B

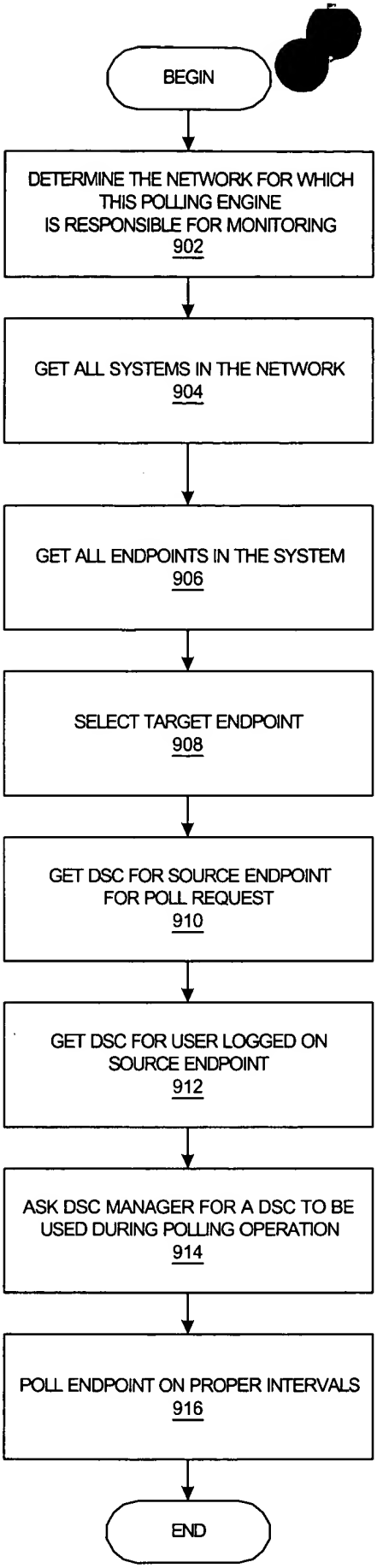

Figure 9A

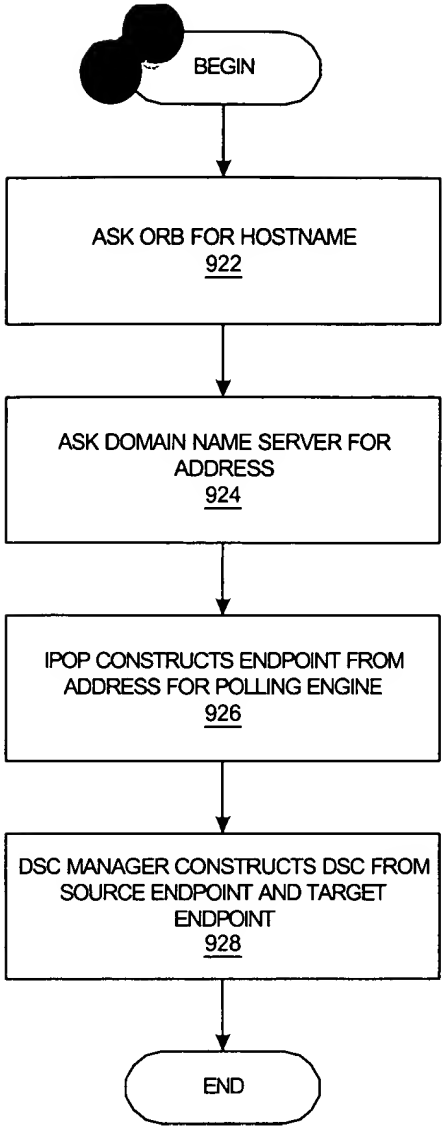

Figure 9B

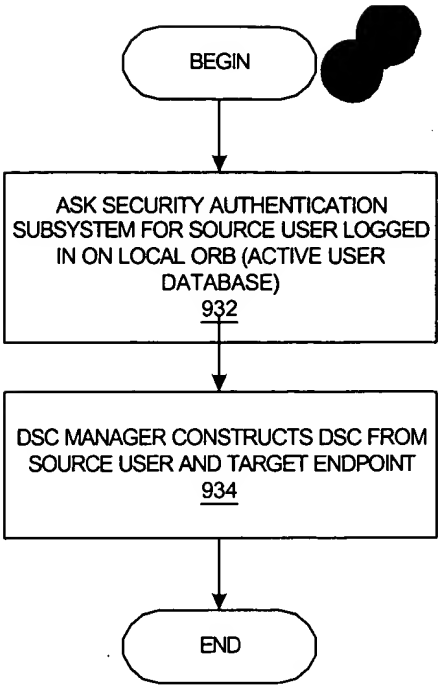

Figure 9C

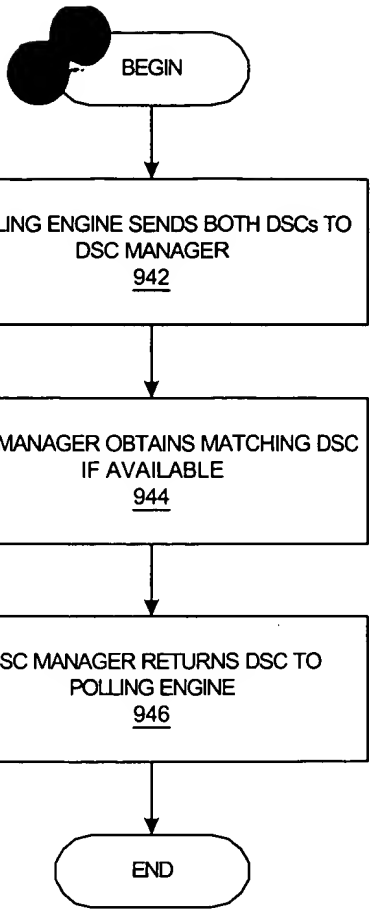

Figure 9D

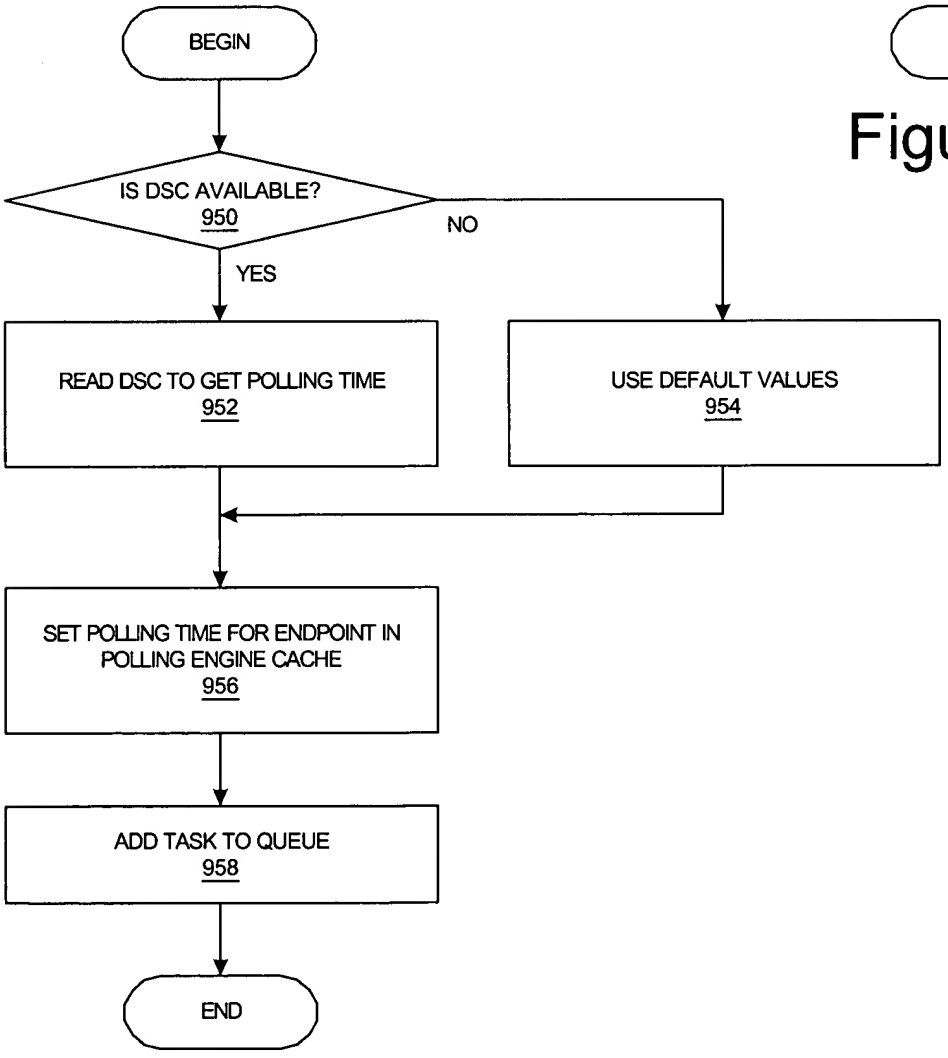

Figure 9E

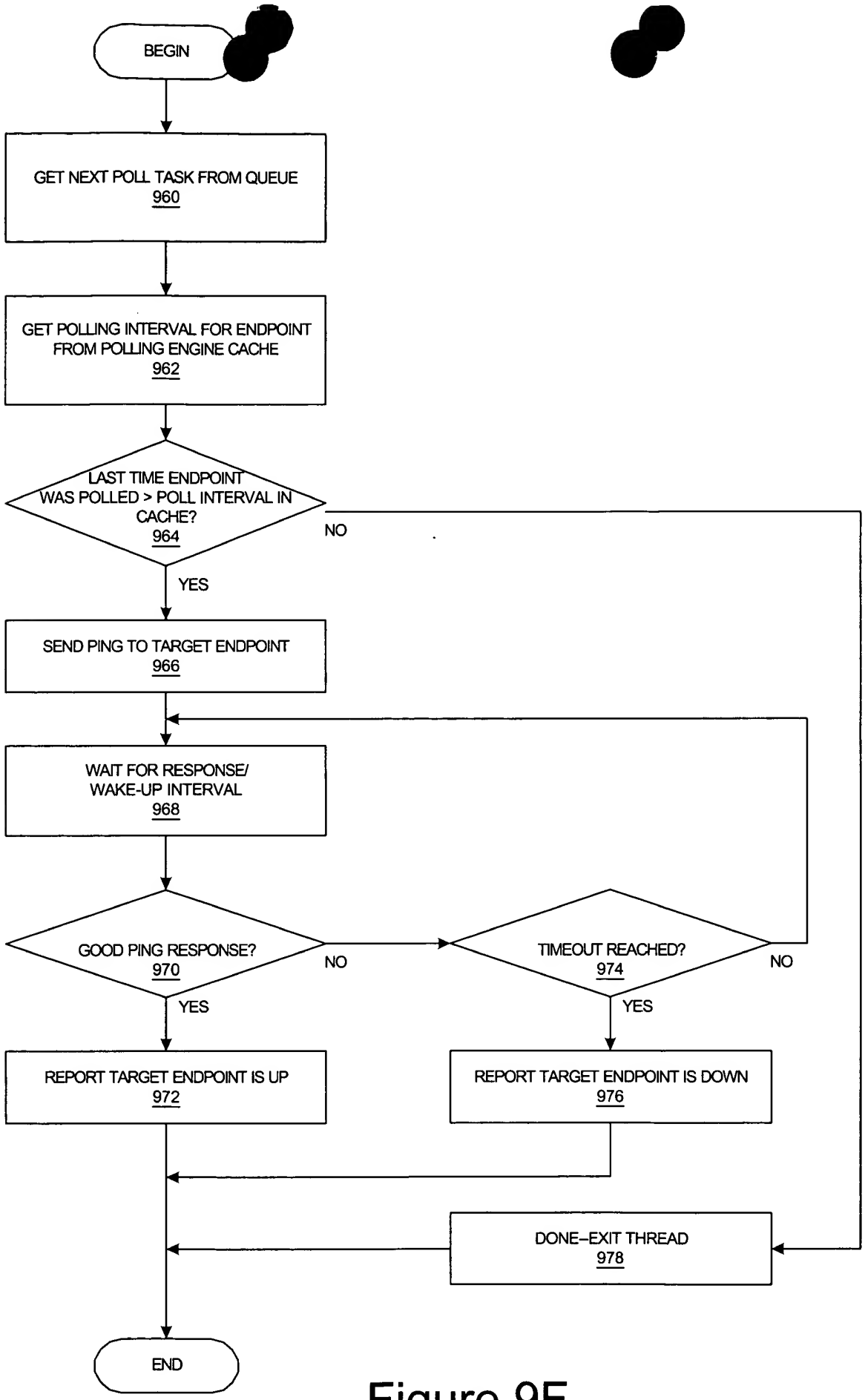
BEGIN
GET NEXT POLL TASK FROM QUEUE
960
GET POLLING INTERVAL FOR ENDPOINT FROM POLLING ENGINE CACHE
962
LAST TIME ENDPOINT WAS POLLED > POLL INTERVAL IN CACHE?
964
NO
YES
SEND PING TO TARGET ENDPOINT
966
WAIT FOR RESPONSE/ WAKE-UP INTERVAL
968
GOOD PING RESPONSE?
970
NO
YES
TIMEOUT REACHED?
974
NO
YES
REPORT TARGET ENDPOINT IS UP
972
REPORT TARGET ENDPOINT IS DOWN
976
DONE–EXIT THREAD
978
END

Figure 9F

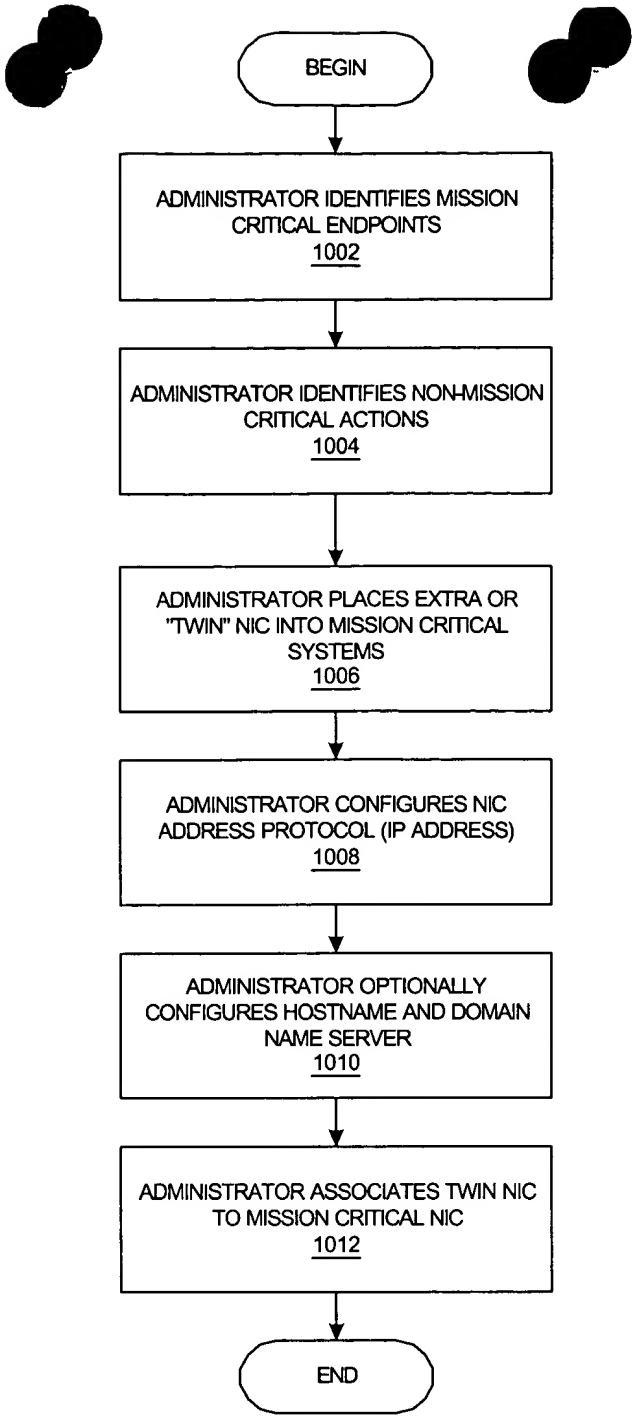
BEGIN
ADMINISTRATOR IDENTIFIES MISSION CRITICAL ENDPOINTS
1002
ADMINISTRATOR IDENTIFIES NON-MISSION CRITICAL ACTIONS
1004
ADMINISTRATOR PLACES EXTRA OR "TWIN" NIC INTO MISSION CRITICAL SYSTEMS
1006
ADMINISTRATOR CONFIGURES NIC ADDRESS PROTOCOL (IP ADDRESS)
1008
ADMINISTRATOR OPTIONALLY CONFIGURES HOSTNAME AND DOMAIN NAME SERVER
1010
ADMINISTRATOR ASSOCIATES TWIN NIC TO MISSION CRITICAL NIC
1012
END

Figure 10A

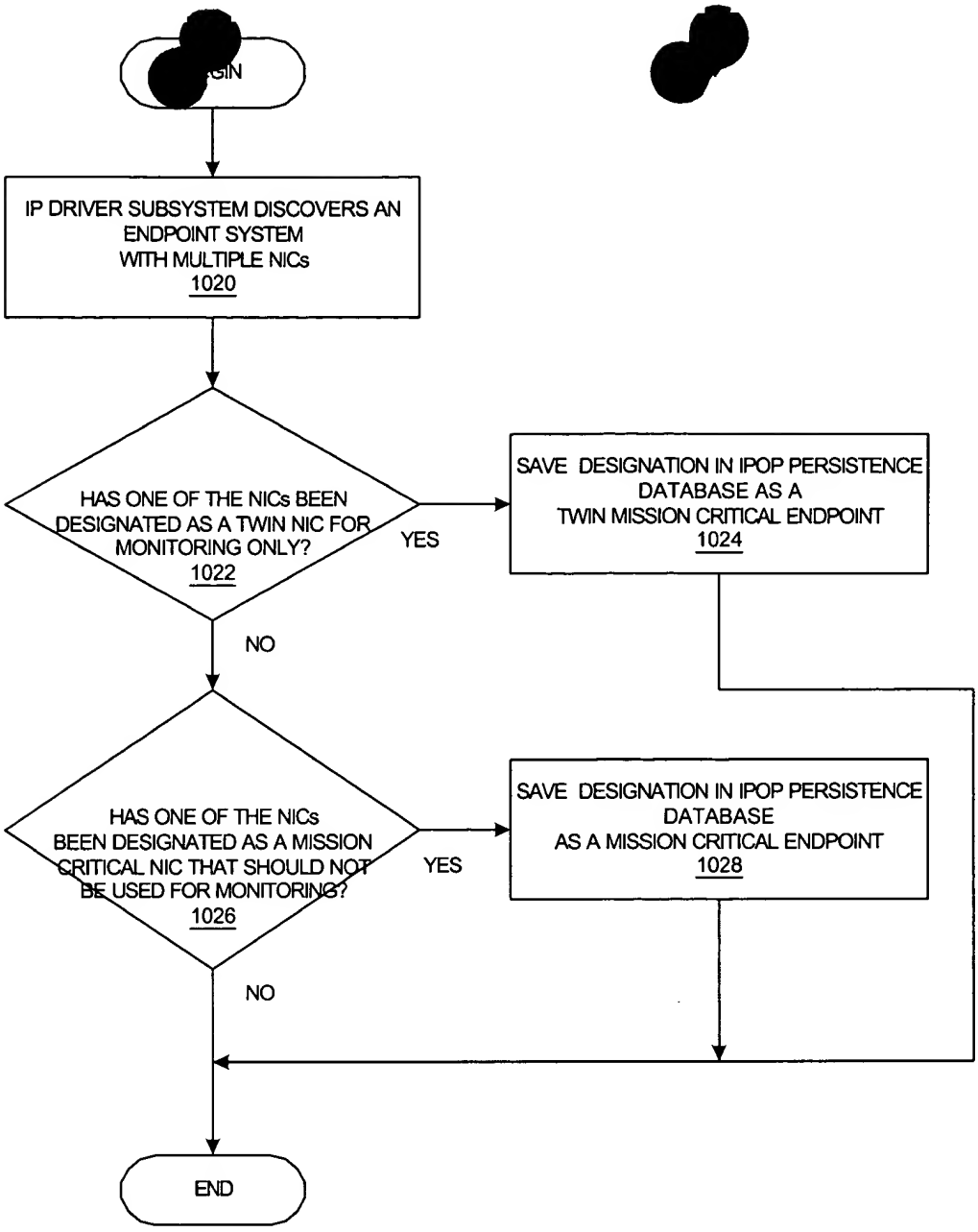
IP DRIVER SUBSYSTEM DISCOVERS AN ENDPOINT SYSTEM WITH MULTIPLE NICs
1020
HAS ONE OF THE NICs BEEN DESIGNATED AS A TWIN NIC FOR MONITORING ONLY?
1022
YES
SAVE DESIGNATION IN IPOP PERSISTENCE DATABASE AS A TWIN MISSION CRITICAL ENDPOINT
1024
NO
HAS ONE OF THE NICs BEEN DESIGNATED AS A MISSION CRITICAL NIC THAT SHOULD NOT BE USED FOR MONITORING?
1026
YES
SAVE DESIGNATION IN IPOP PERSISTENCE DATABASE AS A MISSION CRITICAL ENDPOINT
1028
NO
END

Figure 10B

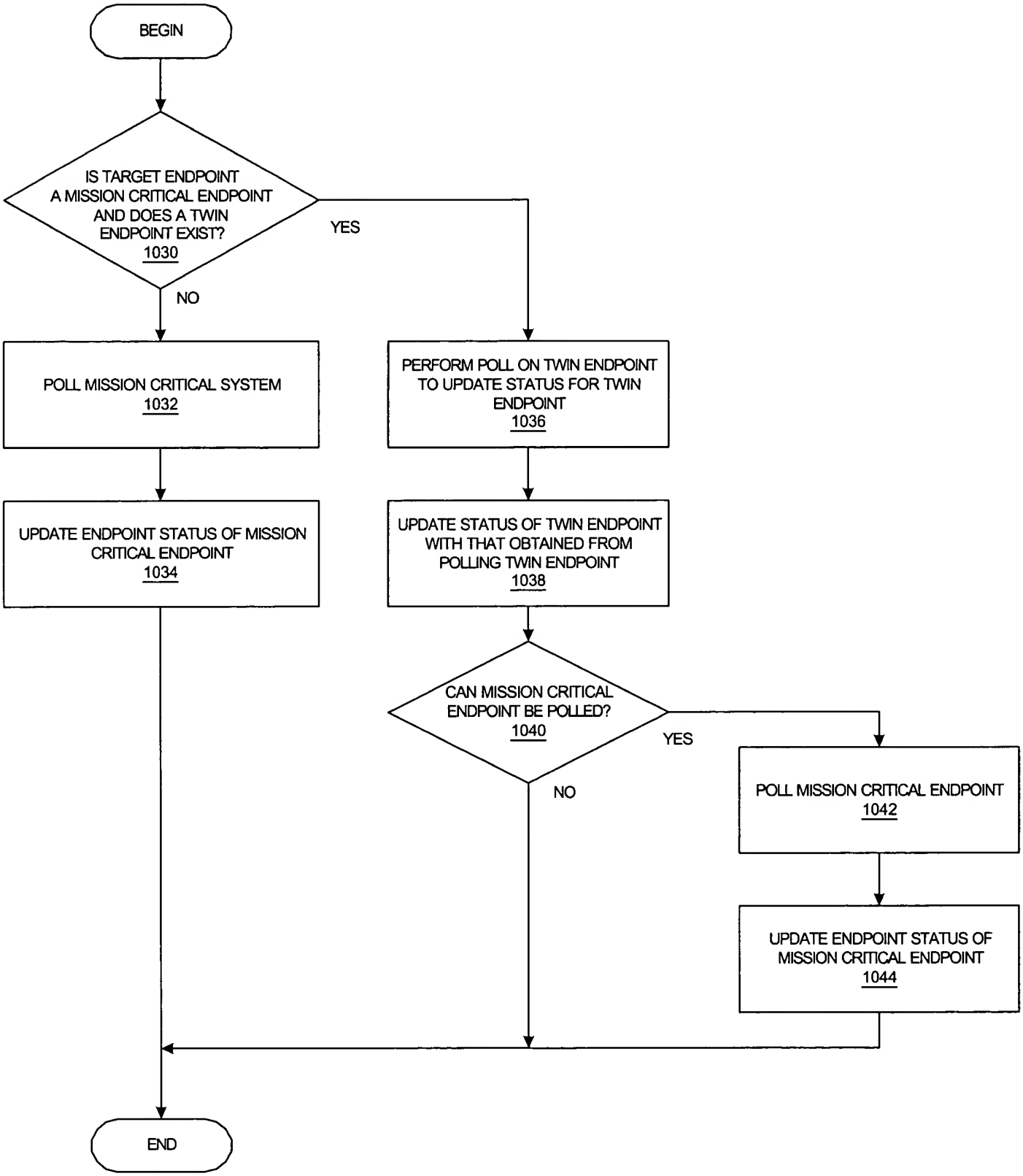
BEGIN
IS TARGET ENDPOINT A MISSION CRITICAL ENDPOINT AND DOES A TWIN ENDPOINT EXIST? 1030
YES
NO
POLL MISSION CRITICAL SYSTEM 1032
UPDATE ENDPOINT STATUS OF MISSION CRITICAL ENDPOINT 1034
PERFORM POLL ON TWIN ENDPOINT TO UPDATE STATUS FOR TWIN ENDPOINT 1036
UPDATE STATUS OF TWIN ENDPOINT WITH THAT OBTAINED FROM POLLING TWIN ENDPOINT 1038
CAN MISSION CRITICAL ENDPOINT BE POLLED? 1040
YES
NO
POLL MISSION CRITICAL ENDPOINT 1042
UPDATE ENDPOINT STATUS OF MISSION CRITICAL ENDPOINT 1044
END

Figure 10C

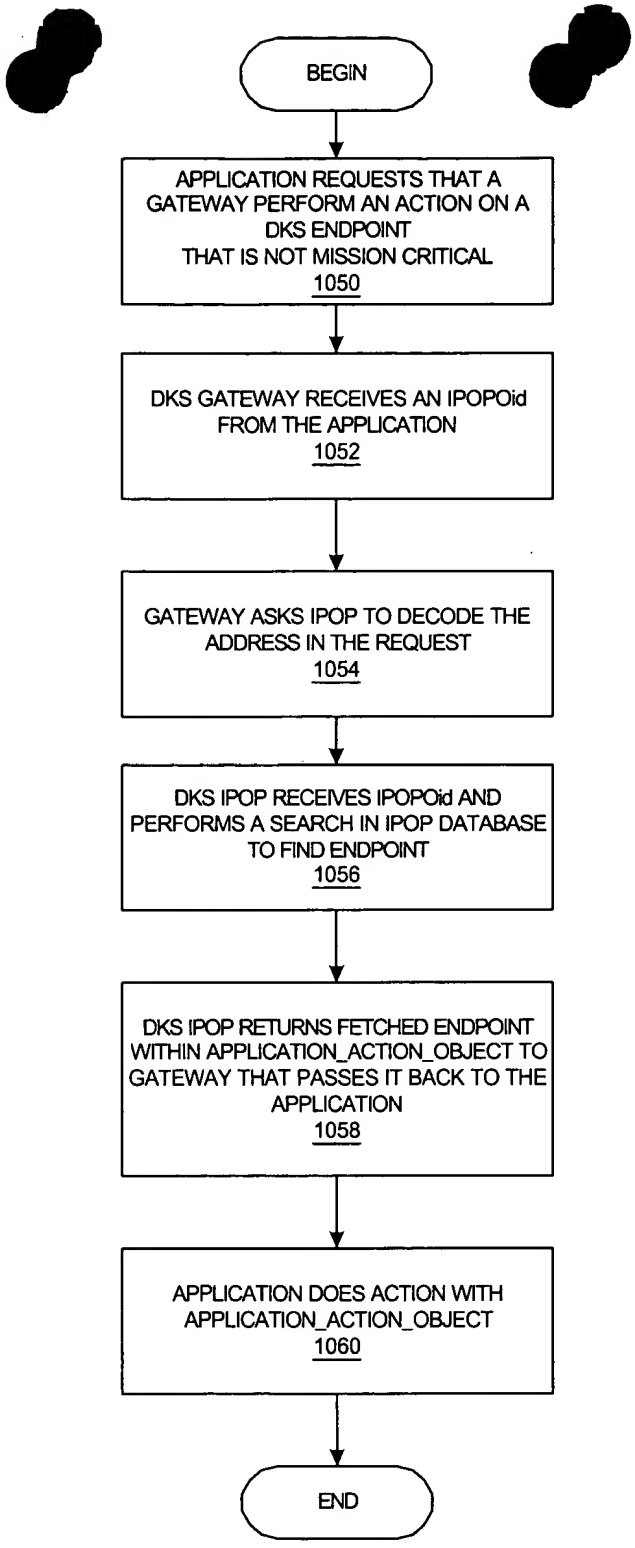
BEGIN
APPLICATION REQUESTS THAT A GATEWAY PERFORM AN ACTION ON A DKS ENDPOINT THAT IS NOT MISSION CRITICAL
1050
DKS GATEWAY RECEIVES AN IPOPOid FROM THE APPLICATION
1052
GATEWAY ASKS IPOP TO DECODE THE ADDRESS IN THE REQUEST
1054
DKS IPOP RECEIVES IPOPOid AND PERFORMS A SEARCH IN IPOP DATABASE TO FIND ENDPOINT
1056
DKS IPOP RETURNS FETCHED ENDPOINT WITHIN APPLICATION_ACTION_OBJECT TO GATEWAY THAT PASSES IT BACK TO THE APPLICATION
1058
APPLICATION DOES ACTION WITH APPLICATION_ACTION_OBJECT
1060
END

Figure 10D

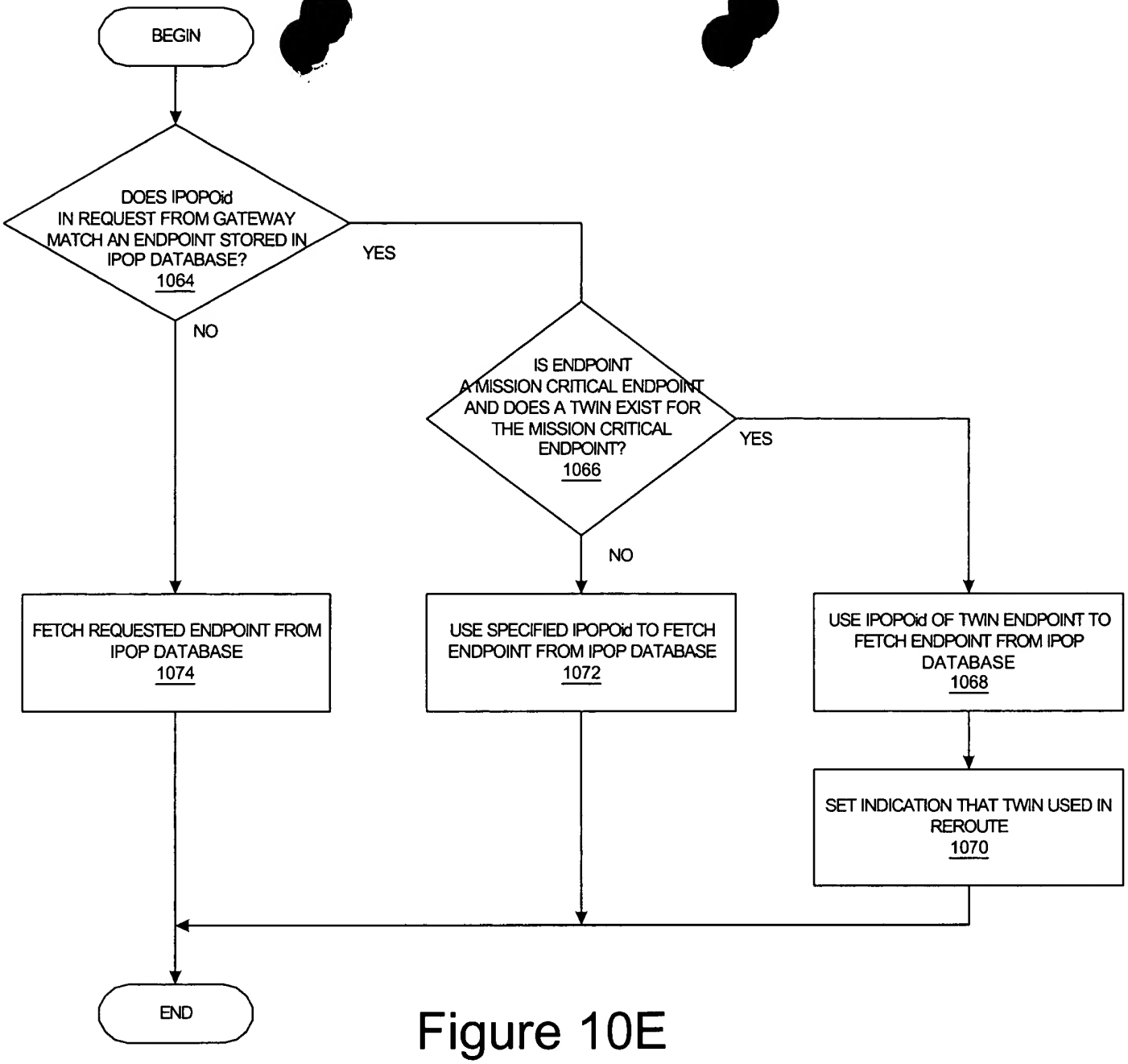

Figure 10E

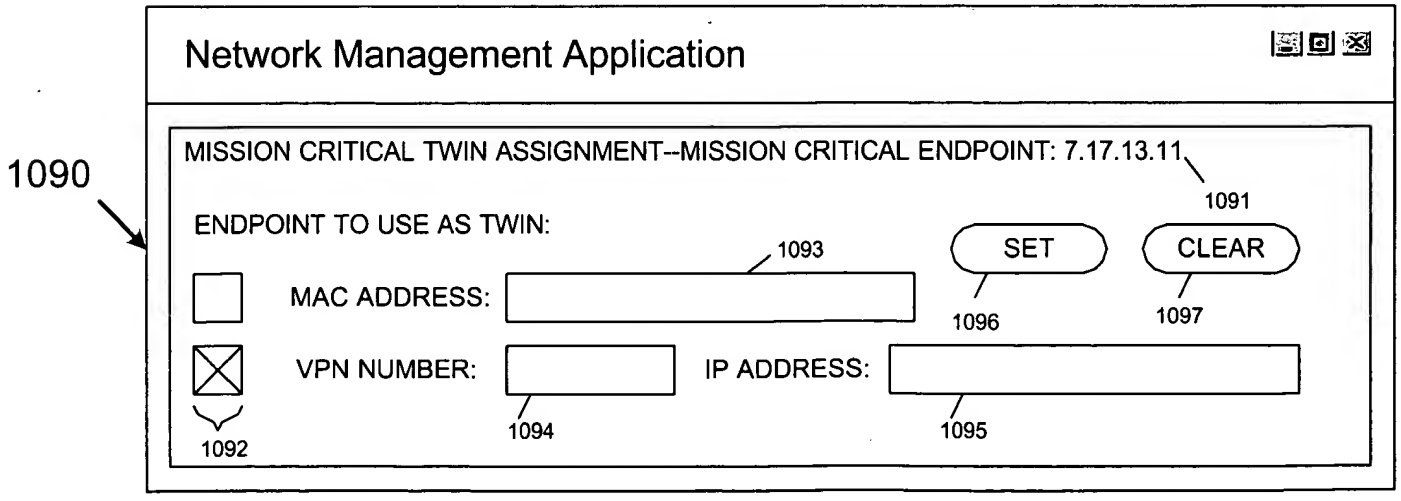

Figure 10F

```
CLASS  ACTION_OBJECT  {

        // CONSTRUCTOR
        ACTION_OBJECT( LONG IPADDRESS, SHORT VIRTUALPRIVATENETWORKADDRESS )
                THROWS BADADDRESS ...
        .
        .
        .
        VOID PERFORMACTION( ) // EXECUTES ACTION METHOD
.
.
.
}
```

Figure 11A

```
CLASS  APPLICATION_ACTION_OBJECT  EXTENDS ACTION_OBJECT {

        boolean IsMissionCriticalAction;  // TRUE = USED TO PERFORM ENTERPRISE-RELATED WORK
                                          // FALSE = USED TO PERFORM MONITORING OPERATIONS
                                          //         OR OTHER NON-REVENUE PRODUCING ACTION

        boolean TwinUsedinReroute         // TRUE = IPOP HAS REPLACED REQUESTED ADDRESS WITH
                                          //         AN ADDRESS THAT CAN BE USED FOR
                                          //         NON-MISSION CRITICAL ACTION
.
.
.
}
```

Figure 11B

```
Public Class Endpoint {

        //public variables
        long                EPObjectID;     //ID to object (both private and public network addresses)
        InetAddress         EPIPAddress;    //physical network address (private or public)
        long                EPVPN;          //virtual private network ID

        //get/set of variables
        public long         getObjectID( ) { ... }
        public InetAddress  getPAddress( ) { ... }
        public long         getVPN( ) { ... }

}
```

Figure 11C

```
Class TwinMissionCriticalEndpoint extends Endpoint {
    .
    .
    .
        IPOPOid     missionCriticalEndpoint; // Mission critical endpoint that is used to gather status
        IPOPOid     missionCriticalSystem;  // Mission critical system

        long        endpointStatus;
        long        twinEndpointStatus;

    .
    .
    .
}
```

Figure 11D

```
Class MissionCriticalEndpoint extends Endpoint {
    .
    .
    .
        boolean     useForMonitoring;       // TRUE = endpoint can be used for monitoring, polling,
                                            //      and other system-management-type resources
    .
    .
    .
}
```

Figure 11E